Лабораторная работа №1

Основы *Git* и *Github*

Цель работы:

Выполнение лабораторной работы направлено на изучение:

1. Наиболее распространенных практик в области контроля версий программного обеспечения, его использования в командной разработке ПО и *DevOps*;

2. Концепции *Git*, основанной на понятиях репозитория и ветвления версий ПО;

3. Порядка использования *GitHub* и его базовых операций.

Порядок работы:

1. Зарегистрировался в *GitHub*, как показано на рисунке 1.

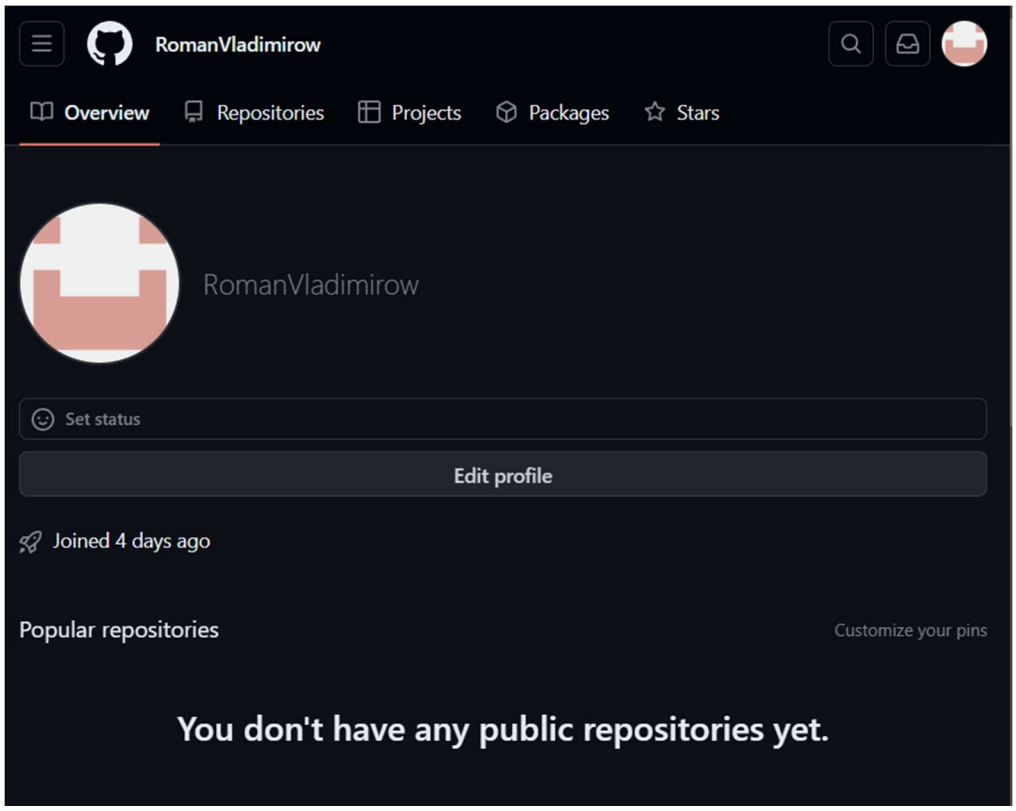

Рисунок 1 - Аккаунт GitHub

2. Создал новый репозиторий: задал имя репозитория, добавил описание, выбрал видимость репозитория «публичный», выбрал опцию «*Initialize this repository with a README*», затем отредактировал файл *README*, как показано на рисунке 2.

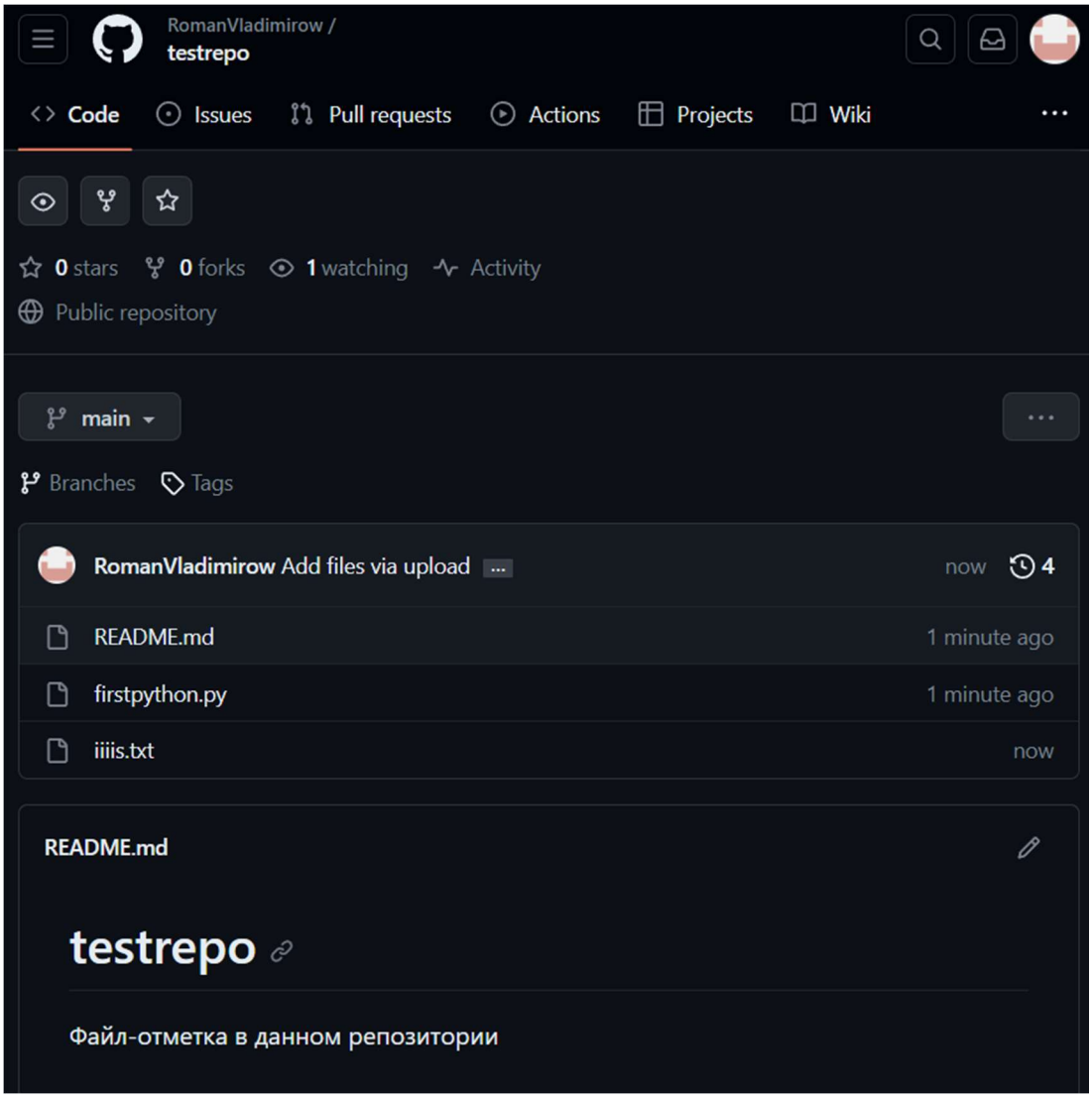

Рисунок 2 - Новый репозиторий

Создал файл *firstpython.py* с помощью встроенного веб-редактора *GitHub*. Зафиксировал изменения в репозитории, как показано на рисунке 3.

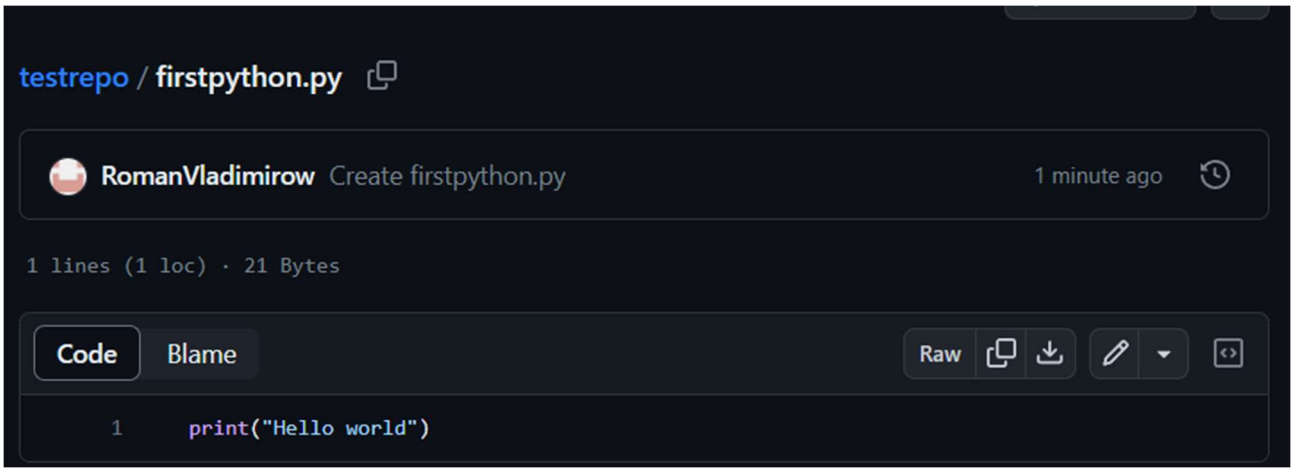

Рисунок 3 - Создание файла

3. Создал новую ветку «*Child_Branch*», создал новый файл. Убедился, что файл, добавленный в дочернюю ветку, не добавляется автоматически в основную ветку, как показано на рисунке 4.

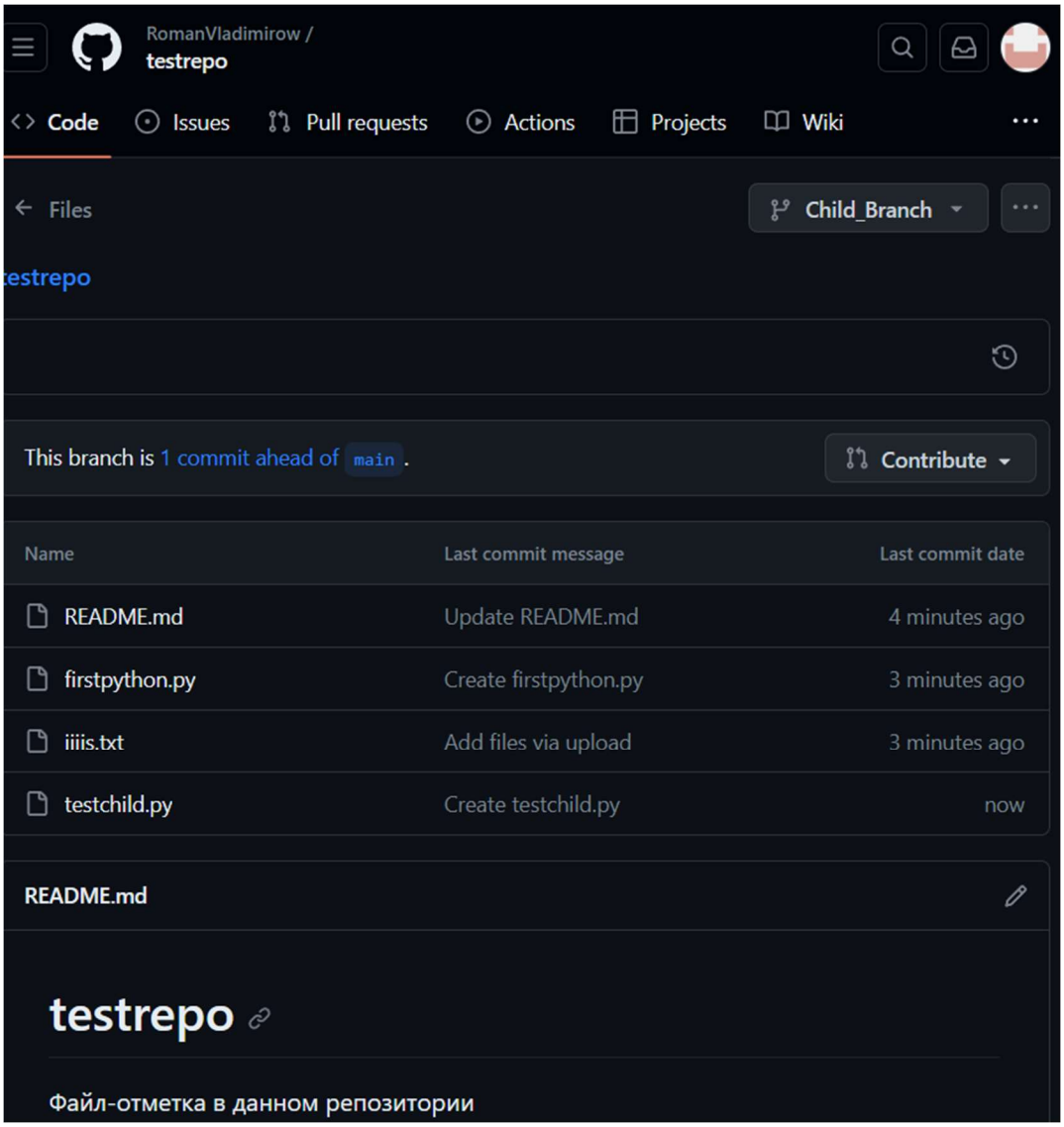

Рисунок 4 - Создание новой ветки

В *Child_Branch* нажал кнопку «*Compare & pull request*», убедился, что в списке указан история ветки, как показано на рисунке 5.

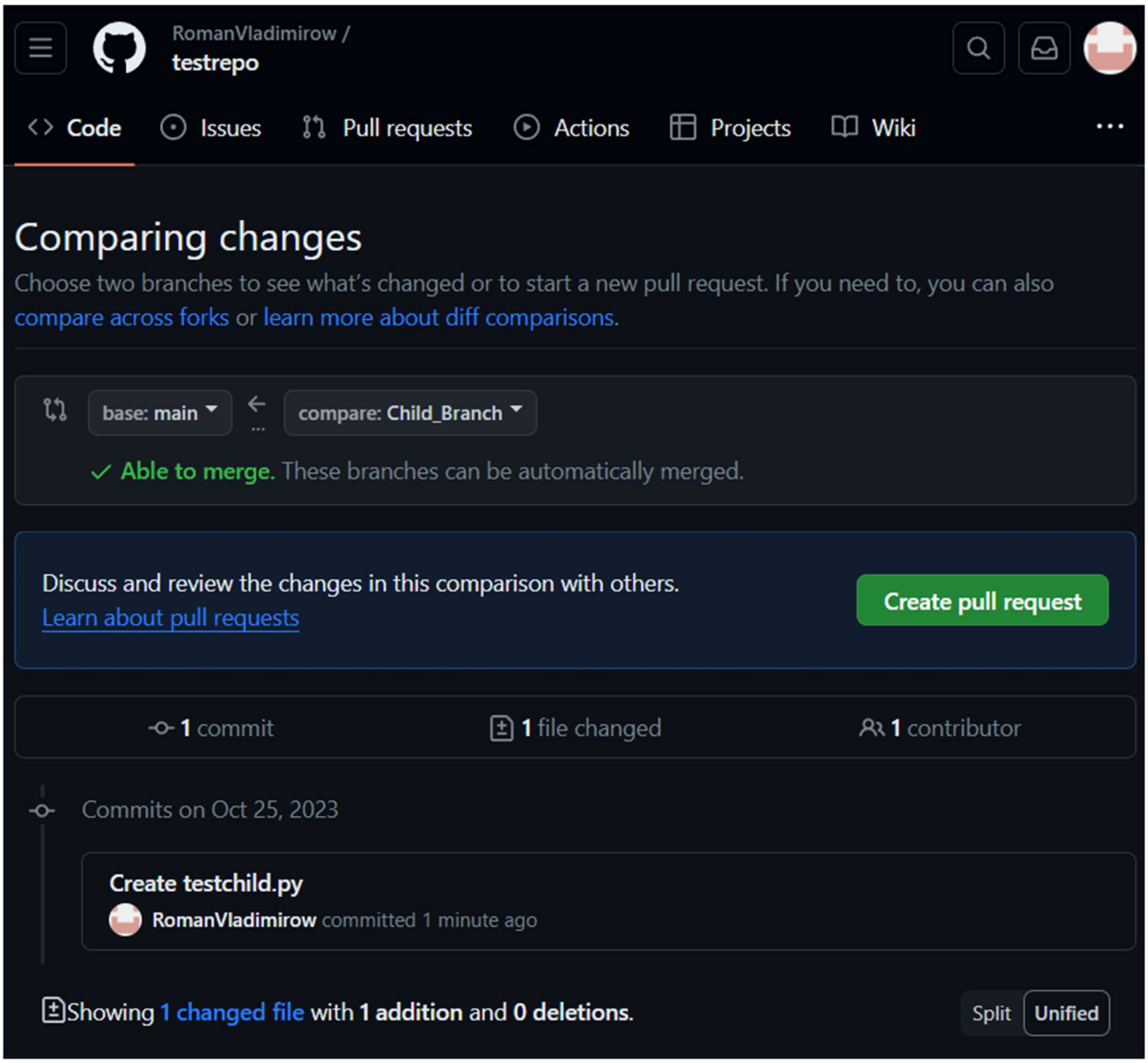

Рисунок 5 - Список измененных файлов

4. Чтобы объединить ветки по запросу *pull request* в проекте, открыл вкладку «*Pull requests*». Отображается список ожидающих запросов на включение. Перешел на нужный *pull request* и нажал «*Merge pull request*», чтобы принять запрос на включение и объединить обновления, как показано на рисунке 6.

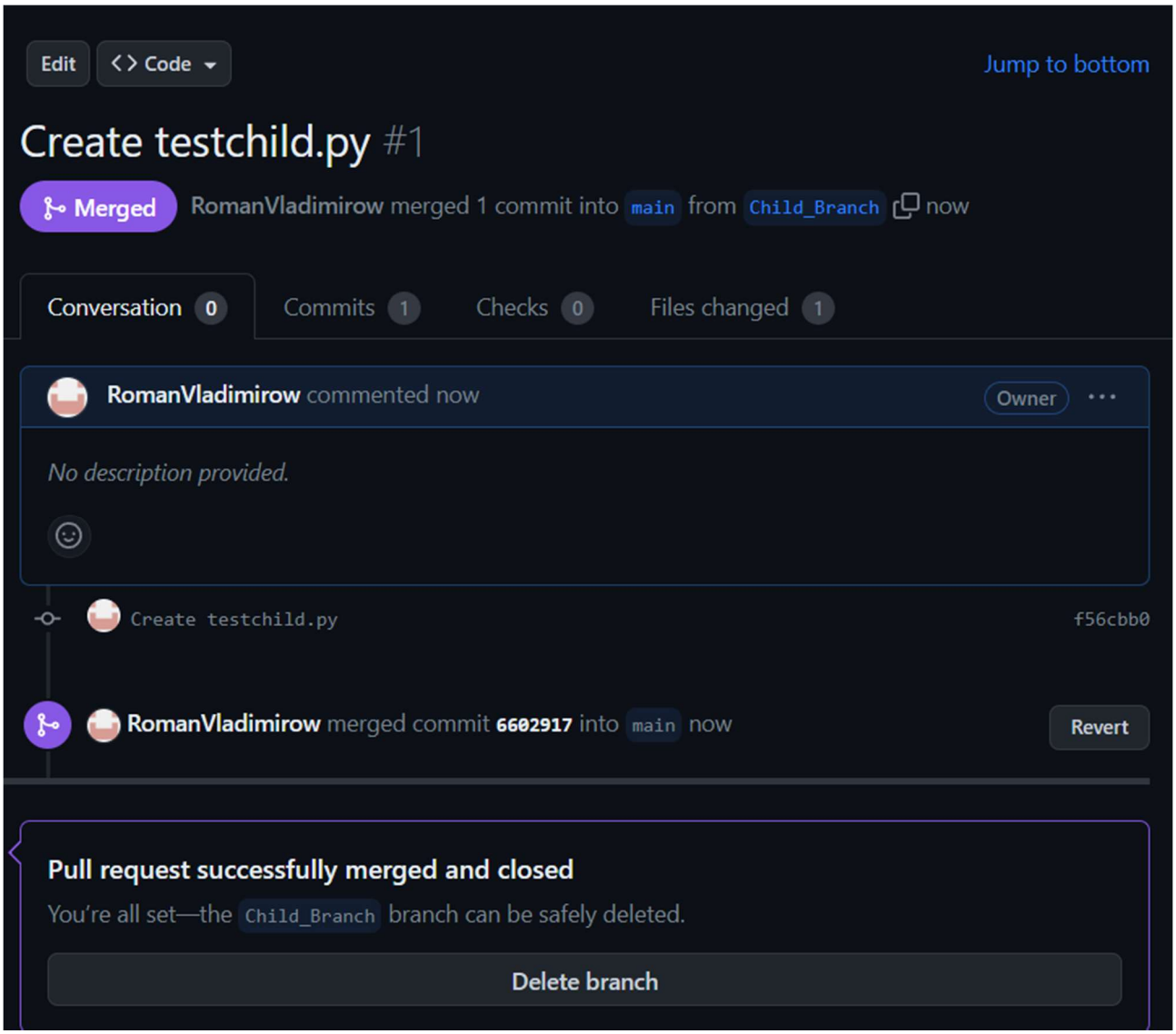

Рисунок 6 - Запрос на включение

5. Произвел работу с локальным репозиторием посредством командной строки *Windows PowerShell* через специальное приложение *Git*.

Создал каталог *myrepo* и перешел в созданный каталог, затем создал локальный репозиторий, вывел на экран содержимое подкаталога *.git*, в котором находится локальный репозиторий, как показано на рисунке 7.

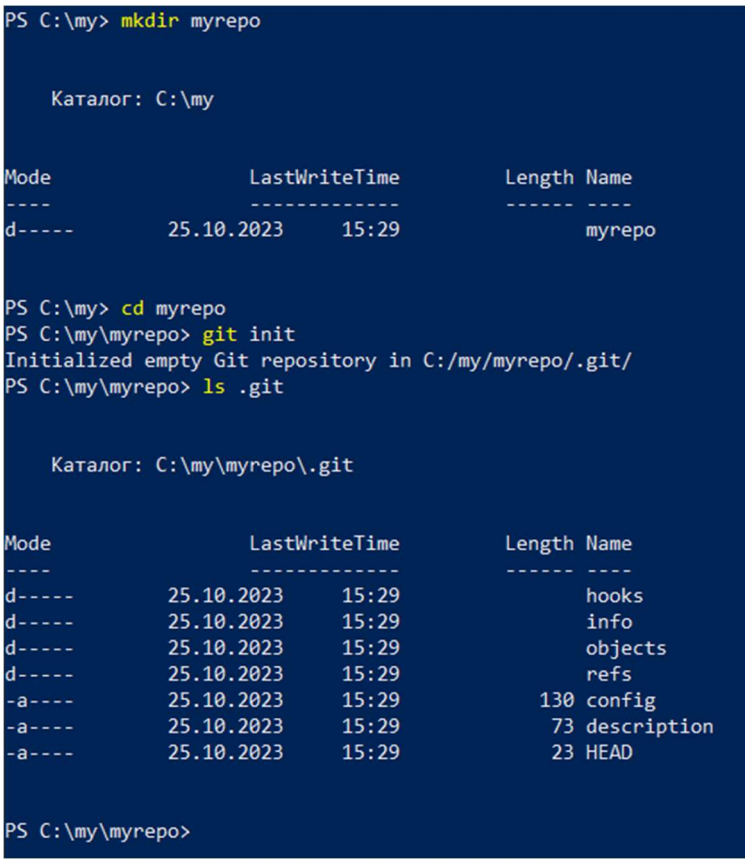

Рисунок 7 - Новый каталог

Создал пустой файл *newfile*, добавил его в репозиторий. Прежде чем зафиксировать изменения, сообщил *git* информацию пользователя. Появившийся в репозитории *newfile* зафиксировал с добавлением сообщения «*added new file*», как показано на рисунке 8.

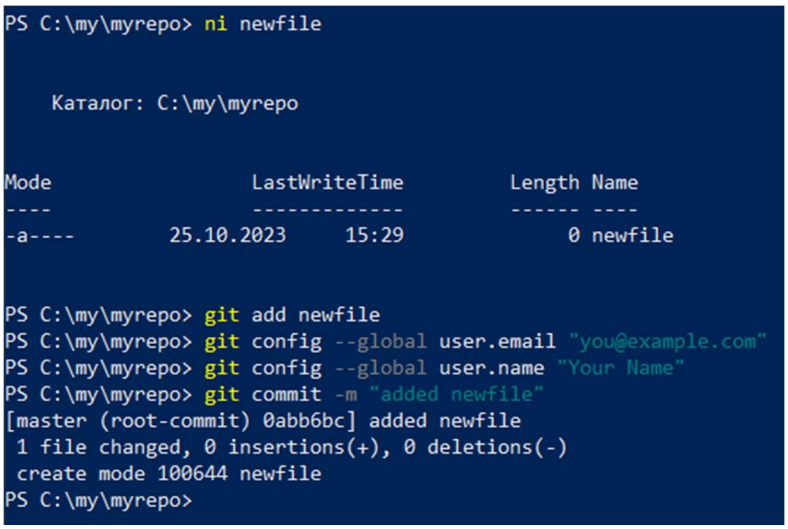

Рисунок 8 - Новый файл

Создал новую ветку под названием *my1stbranch*, убедился в наличии двух веток в репозитории и переключился с ветки *master* на ветку *my1stbranch*, как показано на рисунке 9.

```
PS C:\my\myrepo> git branch my1stbranch
PS C:\my\myrepo> git branch
* master
  my1stbranch
PS C:\my\myrepo> git checkout my1stbranch
Switched to branch 'my1stbranch'
PS C:\my\myrepo> git branch
  master
* my1stbranch
PS C:\my\myrepo>
```

Рисунок 9 - Новая ветка

Внес изменения в *newfile* путем добавления текста, убедился что текст добавлен, как показано на рисунке 10.

```
PS C:\my\myrepo> echo 'New file text.' >> newfile
PS C:\my\myrepo> cat newfile
New file text.
PS C:\my\myrepo>
```

Рисунок 10 - Запись и чтение из файла

Создал новый файл *readme.md* и добавил в репозиторий, проверил изменения в текущей ветке *my1stbranch*, затем добавил *newfile* явно, как показано на рисунке 11.

```
PS C:\my\myrepo> ni readme.md


    Каталог: C:\my\myrepo


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----        25.10.2023     15:31              0 readme.md


PS C:\my\myrepo> git add readme.md
PS C:\my\myrepo> git status
On branch my1stbranch
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   readme.md

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   newfile

PS C:\my\myrepo> git add *
PS C:\my\myrepo> git status
On branch my1stbranch
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   newfile
        new file:   readme.md

PS C:\my\myrepo>
```

Рисунок 11 - Добавление файла

Сохранил изменения в ветку, прикрепив сообщение «*added readme.md modified newfile*», далее получил историю последних коммитов — последний коммит в *my1stbranch*, а также предыдущий коммит в *master*, как показано на рисунке 12.

```
PS C:\my\myrepo> git commit -m "added readme.md modified newfile"
[my1stbranch 2821b62] added readme.md modified newfile
 2 files changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 readme.md
PS C:\my\myrepo> git log
commit 2821b62b582c2d62607f62058c715e29f49d6284 (HEAD -> my1stbranch)
Author: Your Name <you@example.com>
Date:   Wed Oct 25 15:32:56 2023 +0300

    added readme.md modified newfile

commit 0abb6bc8ff6a85d488e71ac406f8a867ed03942e (master)
Author: Your Name <you@example.com>
Date:   Wed Oct 25 15:30:07 2023 +0300

    added newfile
PS C:\my\myrepo>
```

Рисунок 121 - Сохранение изменений

Произвел отмену изменений, использовав ярлык *HEAD* для отката последнего коммита, как показано на рисунке 13.

```
PS C:\my\myrepo> git revert HEAD --no-edit
[my1stbranch dde7bf6] Revert "added readme.md modified newfile"
 Date: Wed Oct 25 15:33:15 2023 +0300
 2 files changed, 0 insertions(+), 0 deletions(-)
 delete mode 100644 readme.md
PS C:\my\myrepo>
```

Рисунок 132 - Отмена изменений

Создал новый *goodfile* и убедился, что файл зафиксирован в *my1stbranch*, как показано на рисунке 14.

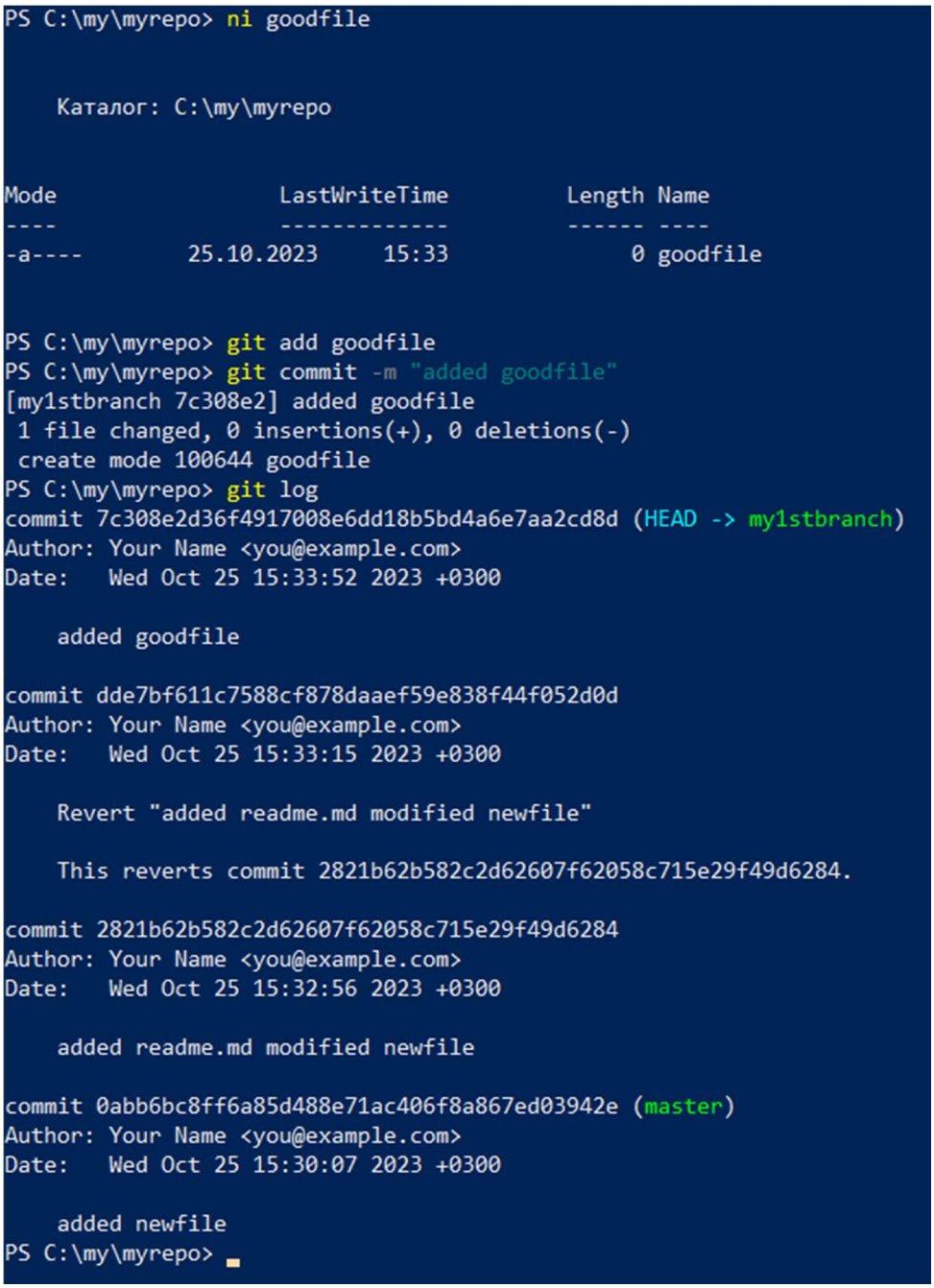

Рисунок 143 - Новый файл

Далее объединил содержимое *my1stbranch* с основной веткой, для этого сначала сделал ветку *master* активной. Произвел слияние веток и вывел на экран журнал. После того, как слияние успешно завершилось, удалил ветку *my1stbranch*, как показано на рисунке 15.

```
PS C:\my\myrepo> git checkout master
Switched to branch 'master'
PS C:\my\myrepo> git merge my1stbranch
Updating 0abb6bc..7c308e2
Fast-forward
 goodfile | 0
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 goodfile
PS C:\my\myrepo> git log
commit 7c308e2d36f4917008e6dd18b5bd4a6e7aa2cd8d (HEAD -> master, my1stbranch)
Author: Your Name <you@example.com>
Date:   Wed Oct 25 15:33:52 2023 +0300

    added goodfile

commit dde7bf611c7588cf878daaef59e838f44f052d0d
Author: Your Name <you@example.com>
Date:   Wed Oct 25 15:33:15 2023 +0300

    Revert "added readme.md modified newfile"

    This reverts commit 2821b62b582c2d62607f62058c715e29f49d6284.

commit 2821b62b582c2d62607f62058c715e29f49d6284
Author: Your Name <you@example.com>
Date:   Wed Oct 25 15:32:56 2023 +0300

    added readme.md modified newfile

commit 0abb6bc8ff6a85d488e71ac406f8a867ed03942e
Author: Your Name <you@example.com>
Date:   Wed Oct 25 15:30:07 2023 +0300

    added newfile
PS C:\my\myrepo> git branch -d my1stbranch
Deleted branch my1stbranch (was 7c308e2).
PS C:\my\myrepo>
```

Рисунок 154 - Слияние веток

Задание №1:

git checkout -b newbranch //Создал новую ветку *newbranch*

ni newbranchfile //Создал пустой файл *newbranchfile*

git add newbranchfile //Добавил файл в свою ветку

git commit -m "добавлен newbranchfile" //Зафиксировал изменения в новой ветке

git revert HEAD –no-edit //Отменил последние зафиксированные изменения

ni newgoodfile //Создал новый файл с именем *newgoodfile*

git add newgoodfile //Добавил последний файл в новую ветку

git commit -m "добавлен newgoodfile" //Зафиксировал изменения

git checkout master `//Переключился на основную ветку`

git merge newbranch `//Объединил изменения в новой ветке с основной`

Результат выполнения представлен на рисунке 16.

```
PS C:\my\myrepo> git checkout -b newbranch
Switched to a new branch 'newbranch'
PS C:\my\myrepo> ni newbranchfile


    Каталог: C:\my\myrepo


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----        25.10.2023     15:35              0 newbranchfile


PS C:\my\myrepo> git add newbranchfile
PS C:\my\myrepo> git commit -m "добавлен newbranchfile"
[newbranch f69a1ad] добавлен newbranchfile
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 newbranchfile
PS C:\my\myrepo> git revert HEAD --no-edit
[newbranch 7b53c58] Revert "добавлен newbranchfile"
 Date: Wed Oct 25 15:35:43 2023 +0300
 1 file changed, 0 insertions(+), 0 deletions(-)
 delete mode 100644 newbranchfile
PS C:\my\myrepo> ni newgoodfile


    Каталог: C:\my\myrepo


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----        25.10.2023     15:35              0 newgoodfile


PS C:\my\myrepo> git add newgoodfile
PS C:\my\myrepo> git commit -m "добавлен newgoodfile"
[newbranch c851c29] добавлен newgoodfile
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 newgoodfile
PS C:\my\myrepo> git checkout master
Switched to branch 'master'
PS C:\my\myrepo> git merge newbranch
Updating 7c308e2..c851c29
Fast-forward
 newgoodfile | 0
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 newgoodfile
PS C:\my\myrepo>
```

Рисунок 165 - Процесс выполнения задания

Задание №2:

Выполнил *fork* проекта второго студента, как показано на рисунке 17.

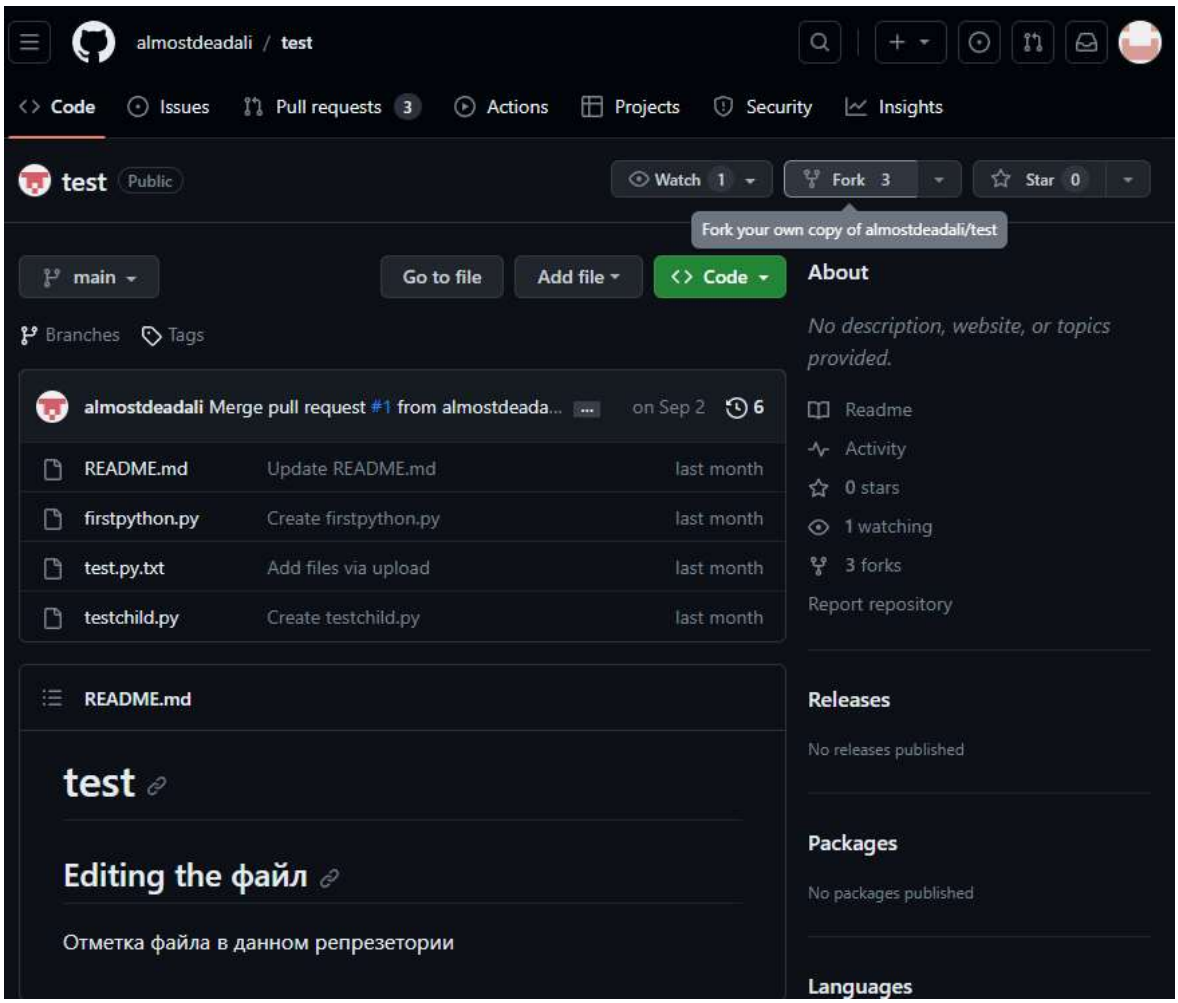

Рисунок 17 - *Fork* проекта второго студента

Выполнил *clone* проекта в локальный репозиторий с помощью *Windows PowerShell*, как показано на рисунке 18.

```
PS C:\my\myrepo> cd C:\my
PS C:\my> mkdir fork


    Каталог: C:\my


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----        25.10.2023     15:38                fork


PS C:\my> cd fork
PS C:\my\fork> clone https://github.com/RomanVladimirow/test.git
clone : Имя "clone" не распознано как имя командлета, функции, файла сценария или выполняемой программы. Проверьте п
равильность написания имени, а также наличие и правильность пути, после чего повторите попытку.
строка:1 знак:1
+ clone https://github.com/RomanVladimirow/test.git
+ ~~~~~
    + CategoryInfo          : ObjectNotFound: (clone:String) [], CommandNotFoundException
    + FullyQualifiedErrorId : CommandNotFoundException

PS C:\my\fork> git clone https://github.com/RomanVladimirow/test.git
Cloning into 'test'...
remote: Enumerating objects: 16, done.
remote: Counting objects: 100% (16/16), done.
remote: Compressing objects: 100% (11/11), done.
remote: Total 16 (delta 2), reused 1 (delta 0), pack-reused 0
Receiving objects: 100% (16/16), done.
Resolving deltas: 100% (2/2), done.
PS C:\my\fork>
```

Рисунок 18 - *Clone* проекта

Добавил новый файл в репозиторий, зафиксировал изменения, как показано на рисунке 19.

```
PS C:\my\fork> cd test
PS C:\my\fork\test> ni windowshell.txt


    Каталог: C:\my\fork\test


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----        25.10.2023     15:40              0 windowshell.txt


PS C:\my\fork\test> echo 'git test' >> windowshell.txt
PS C:\my\fork\test> git add *
PS C:\my\fork\test> git commit -m "добавлен файл"
[main ce00718] добавлен файл
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 windowshell.txt
PS C:\my\fork\test>
```

Рисунок 196 - Новый файл

Выполнил синхронизацию с *fork*-репозиторием, как показано на рисунке 20.

```
PS C:\my\fork\test> git push origin main
info: please complete authentication in your browser...
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 4 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 316 bytes | 316.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/RomanVladimirow/test.git
   095ecf1..ce00718  main -> main
PS C:\my\fork\test>
```

Рисунок 20 - Синхронизация с *fork*-репозиторием

Сформировал *pull request* к *origin* проекту на прием данного изменения, как показано на рисунке 21.

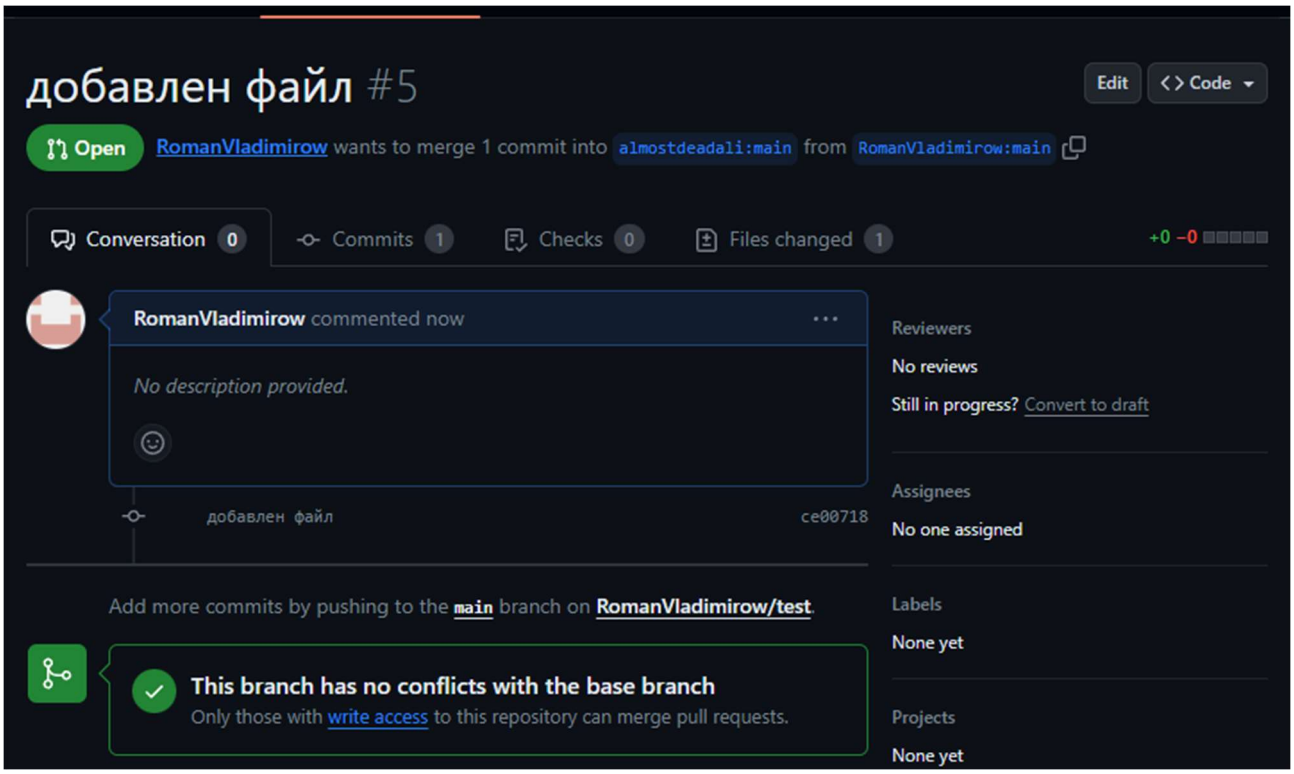

Рисунок 21 - Формирование *pull request*

Вывод: изучили концепцию *Git*, основанную на понятиях репозитория и ветвления версий ПО, изучили порядок использования *GitHub* и его базовых операций, научились работать с онлайн-хостингом *GitHub*.